Analisis Perbandingan Algoritma Klasifikasi Citra Chest X-ray Untuk Deteksi Covid-19

Mohammad Farid Naufal[1*], Selvia Ferdiana Kusuma[2], Kevin Christian Tanus[3], Raynaldy Valentino Sukiwun[4], Joseph Kristiano[5], Jeremy Owen Lieyanto[6], Daniel Cristianindra R.[7]

[1,3,4,5,6]Jurusan Teknik Informatika, Universitas Surabaya, Surabaya, Jawa Timur
[2]Manajemen Informatika, PSDKU Politeknik Negeri Malang, Kediri, Jawa Timur
Email: [1*]faridnaufal@staff.ubaya.ac.id, [2]selvia.ferdiana@polinema.ac.id, [3]s160418043@student.ubaya.ac.id, [4]s160417116@student.ubaya.ac.id, [5]s160418015@student.ubaya.ac.id, [6]s160418081@student.ubaya.ac.id, [7]s160417092@student.ubaya.ac.id

(Naskah masuk: 15 Jan 2021, direvisi: 12 Jun 2021, diterima: 15 Jun 2021)

Abstrak

Kondisi pandemi global Covid-19 yang muncul diakhir tahun 2019 telah menjadi permasalahan utama seluruh negara di dunia. Covid-19 merupakan virus yang menyerang organ paru-paru dan dapat mengakibatkan kematian. Pasien Covid-19 banyak yang telah dirawat di rumah sakit sehingga terdapat data citra *chest X-ray* paru-paru pasien yang terjangkit Covid-19. Saat ini sudah banyak peneltian yang melakukan klasifikasi citra *chest X-ray* menggunakan *Convolutional Neural Network* (CNN) untuk membedakan paru-paru sehat, terinfeksi covid-19, dan penyakit paru-paru lainnya, namun belum ada penelitian yang mencoba membandingkan performa algoritma CNN dan *machine learning* klasik seperti *Support Vector Machine* (SVM), dan *K-Nearest Neighbor (KNN)* untuk mengetahui *gap* performa dan waktu eksekusi yang dibutuhkan. Penelitian ini bertujuan untuk membandingkan performa dan waktu eksekusi algoritma klasifikasi *K-Nearest Neighbors* (KNN), *Support Vector Machine* (SVM), dan CNN untuk mendeteksi Covid-19 berdasarkan citra *chest X-Ray*. Berdasarkan hasil pengujian menggunakan 5 *Cross Validation*, CNN merupakan algoritma yang memiliki rata-rata performa terbaik yaitu akurasi 0,9591, *precision* 0,9592, *recall* 0,9591, dan F1 Score 0,959 dengan waktu eksekusi rata-rata sebesar 3102,562 detik.

Kata Kunci: CNN, SVM, KNN, Chest X-ray, Covid-19

Comparative Analysis of Chest X-ray Image Classification Algorithms for Covid-19 Detection

Abstract

Condition of the global Covid-19 pandemic that emerged at the end of 2019 has become a major problem for all countries in the world. Covid-19 is a virus that attacks the lungs and can cause death. Many Covid-19 patients have been hospitalized so there is chest x-ray image data of patients infected with Covid-19. Currently, there are many studies that classify chest X-ray images using the Convolutional Neural Network (CNN) to distinguish healthy lungs, Covid-19 infection, and other lung diseases, but no research has attempted to compare the CNN algorithm classical machine learning algorithm such as the K-Nearest Neighbor (KNN) and Support Vector Machine (SVM) to see the performance and execution time gap. This study aims to compare the performance and execution time of the K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and CNN algorithms for detecting Covid-19 based on X-Ray chest images. Based on the test results using 5 Cross Validation, CNN is the algorithm that has the best performance, namely 0.9591 accuracy, 0.9592 precision, 0.9591 recall, and F1 Score 0.959 with an average execution time of 3102.562 seconds.

Keywords: *CNN, SVM, KNN, Chest X-ray, Covid-19.*

I. PENDAHULUAN

Tahun 2019 merupakan awal mula dari munculnya kasus *Corona Virus Disease* – 19, yang disebut juga dengan Covid-19, pertama kali muncul di Wuhan, China. Semenjak saat itu, lonjakan orang yang terinfeksi Covid-19 mulai melonjak yang tadinya hanya berjumlah puluhan orang yang kemudian persebaran virus ini terus naik hingga akhirnya Covid-19 ini terus tersebar hingga ke penjuru dunia. Dalam perkembangannya, tingkat penularan Covid-19 juga dinilai cukup tinggi. Sampai saat ini, di bulan Januari 2021, di saat penulisan artikel ini ditulis, total kasus Covid-19 di dunia sudah menginjak hingga kurang lebih 86.248.818 kasus positif, dengan total 61.197.913 pasien sembuh, dan total kasus kematian hingga 1.863.861 kasus [1].

Shi, et.al. [2] melakukan analisis deskriptif terhadap 81 pasien Covid-19 menggunakan *radiological* CT dan menyatakan bahwa citra *Chest* CT dan *X-Ray* dapat secara efektif mendeteksi Covid-19 pada pasien dengan gejala. Islam et.al. [3] menyatakan bahwa diagnosis Covid-19 menggunakan citra *Chest X-ray* dapat digunakan sebagai tes tambahan jika pasien dinyatakan negatif pada tes PCR namun memiliki gejala. Yasin, et.al. [4] melakukan *monitoring* citra *Chest X-ray* pada pasien Covid-19 dan memberikan kesimpulan bahwa citra *Chest X-ray* dapat digunakan untuk *monitoring* jangka panjang pasien. Dari beberapa penelitian tersebut dapat dikatakan bahwa citra *Chest X-ray* adalah *dataset* yang berguna bagi tenaga medis untuk menangani pasien Covid-19.

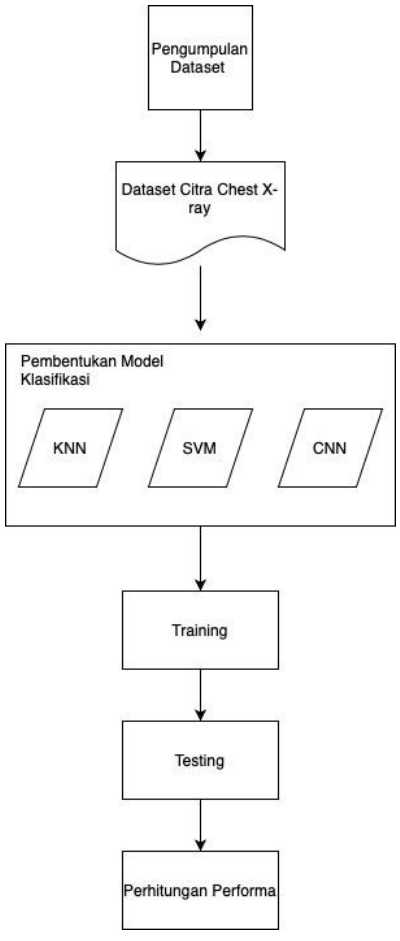
Gao, et.al. [5] menggunakan CNN dengan arsitektur VGG-19 dalam melakukan klasifikasi citra *Chest X-Ray* ke dalam 3 kategori yaitu normal, *covid*, dan *bacterial pneumonia*. Akurasi yang dihasilkan 95%. Abbas, et.al. [6] menggunakan *Decompose*, *Transfer*, *Compose* (*DeTraC*) CNN untuk klasifikasi *Chest X-Ray* menjadi 3 kategori, yaitu normal, Covid-19, dan SARS. Akurasi yang dihasilkan mencapai 93,1%. Deng, et.al. [7] menggunakan *pre-trained model* Keras pada data *Chest X-ray* dan *Chest CT*. Akurasi yang dihasilkan dikedua *dataset* tersebut adalah 84% dan 75%. Performa yang dihasilkan tersebut masih memiliki akurasi di bawah 90%. Dari beberapa penelitian yang disebutkan belum ada penelitian yang mencoba untuk membandingkan performa algoritma *machine learning* klasik seperti K-*Nearest Neighbors*, *Support Vector Machine* (SVM), dengan algoritma *Neural network* yaitu *Convolutional Neural Network* (CNN) dalam melakukan deteksi Covid-19 pada citra *Chest X-ray*.

Berdasarkan penelitian yang sudah dilakukan sebelumnya, belum terdapat penelitian yang bertujuan untuk membandingkan performa algoritma klasifikasi pada *dataset Chest X-ray* yang sama. Selain itu belum ada penelitian yang membahas tentang hubungan antara waktu komputasi dan performa algoritma klasifikasi. Penelitian ini bertujuan untuk membandingkan performa beberapa algoritma dalam melakukan klasifikasi terhadap citra *Chest X-ray* ke dalam kategori paru-paru yang sehat, paru-paru yang terjangkit Covid-19, serta paru-paru yang terjangkit penyakit lainnya, dalam kasus ini adalah viral *pneumonia*. Algoritma yang dibandingkan dalam penelitian ini adalah KNN, SVM, dan CNN. Fitur yang digunakan dalam melakukan klasifikasi adalah intensitas *pixel*. Performa yang dibandingkan antara lain akurasi, *precision*, *recall*, *f1 score*, dan waktu eksekusi. Perbandingan performa algoritma ke depannya diharapkan dapat berguna untuk peneliti dalam memilih algoritma untuk melakukan klasifikasi citra *Chest X-ray*. Tujuan aplikatif penelitian ini adalah mengetahui perbandingan algoritma klasifikasi yang akan memudahkan peneliti dan praktisi dalam memilih algoritma yang paling tepat untuk klasifikasi Citra *Chest X-Ray* untuk deteksi Covid-19.

Sistematika penelitian ini terdiri dari 4 bagian. Pada bab 1 dijelaskan mengenai latar belakang masalah penelitian. Pada bab 2 dijelaskan mengenai metodologi penelitian. Pada bab 3 dijelaskan mengenai hasil dan pembahasan. Pada bab 4 dijelaskan mengenai kesimpulan.

II. METODOLOGI PENELITIAN

Metodologi penelitian yang digunakan terdiri dari 5 tahapan, yaitu pengumpulan *dataset*, pembentukan model klasifikasi, *training* model klasifikasi, *testing* model klasifikasi, dan perhitungan performa. Gambar 1 menunjukkan alur metodologi penelitian yang digunakan.

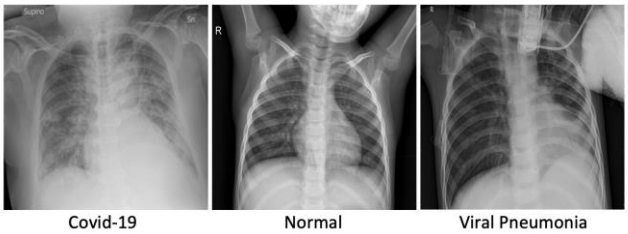


Gambar 1. Metodologi Penelitian

DOI: 10.34148/teknika.v10i2.331

A. Pengumpulan *Dataset*

Dataset yang digunakan dalam penelitian ini adalah citra *Chest X-ray* yang didapatkan dari Kaggle [8]. Tujuan penggunaan *dataset* dari Kaggle adalah *dataset* ini dapat diakses publik sehingga dapat dibandingkan dengan penelitian selanjutnya. *Dataset* terdiri dari 3 jenis atau label citra *Chest X-ray*, yaitu paru-paru normal, terinfeksi Covid-19, dan viral *pneumonia*. Gambar 2 menunjukkan contoh citra *Chest X-ray* paru-paru normal, terinfeksi Covid-19, dan viral *pneumonia*. Tabel 1 menununjukkan detail jumlah *dataset* dari tiap jenis Citra *Chest X-ray*. Dikarenakan *dataset* citra *Chest X-ray* memiliki ukuran citra yang berbeda-beda, maka semua data citra *Chest X-ray* dilakukan *resize* dengan ukuran 64x64.



Gambar 2. Contoh Citra *Chest X-ray*

Tabel 1. Detail Jumlah *Dataset Citra Chest X-ray*

Jenis	Jumlah
Normal	1341
Covid-19	1200
Viral *Pneumonia*	1345

B. Pembentukan Model Klasifikasi

Pada tahapan ini dilakukan konfigurasi berbagai parameter yang digunakan dari algoritma KNN, SVM, dan CNN. Tujuannya adalah agar mengetahui pengaruh dari parameter terhadap performa algoritma.

1) KNN

Konfigurasi parameter yang digunakan pada algoritma KNN adalah jumlah *Neighbors* dan jenis *distance*. Tabel 2 menunjukkan konfigurasi parameter yang digunakan dalam algoritma KNN. Jumlah *Neighbors* yang digunakan adalah 5, 7, dan 9. Sedangkan jenis *distance* yang digunakan adalah *Euclidean*, *Manhattan*, dan *Minkowski*. Setiap *distance* dilakukan uji coba dengan jumlah *neighbors* sebanyak 5, 7, dan 9.

Tabel 2. Parameter KNN

Parameter	Deskripsi
Jumlah *Neighbors*	5, 7, 9
Distance	*Euclidean, Minkowski*

2) SVM

Konfigurasi parameter yang digunakan pada algoritma SVM adalah jenis kernel. Tabel 3 menunjukkan konfigurasi parameter yang digunakan dalam algoritma SVM. Jenis kernel yang digunakan adalah *Linear*, *Poly*, dan RBF. Pada tahapan *training* akan dipilih salah satu kernel yang memiliki performa terbaik.

Tabel 3. Parameter KNN

Parameter	Deskripsi
Kernel	*Linear, Poly,* RBF

3) CNN

Konfigurasi parameter yang digunakan pada algoritma CNN adalah jumlah *epoch*, model *convolution*, jenis *activation function*, jumlah *dense layer*, dan jumlah *batch size*. Tabel 4 menunjukkan parameter CNN yang digunakan.

Tabel 4. Parameter CNN

Parameter	*Output Shape*	Deskripsi
conv2d	(*None*, 62, 62, 32)	*Filter_size* = 3x3. *Act = ReLu*
(*MaxPooling2D*)	(*None*, 31, 31, 32)	*Pool_size* = 2
(*Conv2D*)	(*None*, 29, 29, 32)	*Filter_size* = 3x3. *Act = ReLu*
(*MaxPooling2D*)	(*None*, 14, 14, 32)	*Pool_size* = 2
(*Flatten*)	(*None*, 6272)	-
(*Dense*)	(*None*, 128)	*Act = ReLu*
(*Dense*)	(*None*, 3)	*Act= Softmax*
epoch	-	50
optmizer	-	Adam
Batch_size	-	8

Activation function pada dua tahapan *convolution* dan *dense* pada *layer* pertama adalah *Rectified Linear Unit* (*ReLu*). Rumus *activation function ReLu* dapat dilihat pada persamaan (1). *x* adalah nilai yang dimasukkan ke *activation function.*

$$R(x) = max(0, x) \quad (1)$$

Sedangkan *activation function* pada *dense layer* kedua adalah *softmax* dikarenakan terdapat 3 label klasifikasi. Rumus *activation function softmax* dapat dilihat pada persamaan (2). x_i adalah nilai *input* yang berasal dari *layer* sebelumnya, n adalah jumlah label, dan j adalah urutan label.

$$S(x_i) = \frac{e^{x_i}}{\sum_j^n e^{x_j}} \quad (2)$$

Epoch yang digunakan saat proses *training* adalah 50. *Optimizer* untuk memperbarui bobot *edge* pada tiap *layer* adalah *Adam optimizer*. *Batch size* yang digunakan adalah 8.

4) Spesifikasi Perangkat Keras

Semua model klasifikasi yang dibentuk, proses *training* dan *testing* dieksekusi pada sebuah komputer dengan spesifikasi pada Tabel 5. Parameter Spesifikasi yang ditampilkan adalah CPU, RAM, *Space of Disk*, dan GPU *Model Name*. Informasi spesifikasi ini penting dikarenakan berpengaruh pada performa waktu eksekusi yang diteliti.

Tabel 5. Spesifikasi Perangkat Keras

Parameter	Spesifikasi
CPU	*Intel® Core™* i3-3220 CPU @ 3.30GHz (4 CPUs)
RAM	8192 MB
Operating System	*Windows* 10 *Education* 64-bit (10.0, Build 17763)
GPU *Model Name*	AMD *Radeon* HD 5500 *Series*
GPU *Total Memory*	1 GB

C. *Training*

Pada tahapan *training* dan *testing*, *dataset* yang digunakan dibagi menjadi dua bagian dengan proporsi 80% *training* dan 20% *testing*, sehingga jumlah *cross validation* yang digunakan adalah 5. Pembagian *data training* dan *testing* tersebut dilakukan secara acak.

Training dilakukan untuk algoritma KNN, SVM, dan CNN dengan menggunakan parameter yang telah dijelaskan pada subbab sebelumnya. Pada algoritma CNN, terdapat tahapan data *augmentation* untuk memperkaya variasi dari *dataset*. Semakin bervariasi *dataset training* berguna untuk menghindari terjadinya *overfitting*. Jenis data *augmentation* yang digunakan adalah *horizontal flip*, *shear range*, dan *zoom range*. Data *augmentation* dilakukan menggunakan *library* Keras [9].

Horizontal flip digunakan menduplikasi *dataset training* dengan cara merotasi citra sebesar 90 derajat. *Shear range* [10] melakukan *shear transformation* yang berguna untuk merotasi citra dengan derajat tertentu sesuai dengan parameter. *Zoom range* digunakan untuk memperbesar citra dengan ukuran tertentu sesuai dengan parameter.

D. *Testing*

Tahapan *testing* dilakukan untuk memvalidasi model yang telah terbentuk pada tahapan *training*. *Testing* dilakukan di setiap *cross validation* untuk algoritma KNN, SVM, dan CNN. *Testing* menggunakan *cross validation* bertujuan untuk melihat apakah model yang dibangun di algoritma memiliki performa yang stabil atau tidak.

Pada algoritma CNN, validasi dilakukan di setiap *epoch* menggunakan *data testing*. Jika terdapat model di *epoch* tertentu yang memiliki performa terbaik, maka model tersebut akan disimpan. Penyimpanan model terbaik menggunakan *checkpoint* yang merupakan *library* di Keras.

E. Perhitungan Performa

Pada tahapan ini dilakukan proses perhitungan performa dari tahapan *testing* di setiap algoritma. *Metric* performa yang digunakan adalah *accuracy*, *precision*, *recall*, *f1 score*, dan waktu eksekusi proses *training* dan *testing*. Setiap algoritma dengan parameternya masing-masing dihitung performanya di setiap *cross validation* dan kemudian dihitung rata-ratanya. Parameter disebuah algoritma yang memiliki performa terbaik akan dipilih dan kemudian dibandingkan dengan algoritma yang lain yang memiliki parameter dengan performa yang terbaik.

Persamaan (3) menunjukkan perhitungan *accuracy*. *Accuracy* digunakan untuk menghitung total dari *True Positive* (TP) dan *True Negative* (TN) dibagi dengan total dari TP, TN, *False Positive* (FP), dan *False Negative* (FN).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (3)$$

Persamaan (4) menunjukkan rumus perhitungan dari *Precision*. *Precision* dihitung dengan cara membagi TP dengan total dari TP dan FP.

$$Precision = \frac{TP}{TP+FP} \quad (4)$$

Persamaan (5) menunjukkan rumus perhitungan dari *recall*. *Recall* dihitung dengan cara membagi anatara TP dengan total dari TP dan FN.

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

Persamaan (6) menunjukkan rumus perhitungan dari *f1 Score*. *F1 Score* dihitung dengan cara membagi antara perkalian *Precision* dengan *Recall* dan penambahan *Precision* dan *Recall*.

$$F1\ Score = \frac{Precision\ x\ Recall}{Precision+Recall} \quad (6)$$

Perhitungan rumus *precision*, *recall*, dan *f1 score* pada kasus ini menggunakan *weighted metric*. *Weighted metric* digunakan untuk kasus klasifikasi *multiclass*. *Dataset* citra *Chest X-ray* yang digunakan pada kasus ini memiliki jumlah yang berbeda tiap kelas klasifikasinya seperti yang ada pada Tabel 1. Rumus perhitungan *weighted metric* dapat dilihat pada Persamaan (7). p_i adalah *metric performa precision*, *recall*, dan *f1 score* untuk masing-masing kelas *i*. c_i adalah jumlah *dataset* pada masing-masing kelas *i*.

$$W_m = \frac{\sum_i^j p_i c_i}{\sum_i^j c_i} \quad (7)$$

III. HASIL DAN PEMBAHASAN

Pada tahapan ini akan dijelaskan mengenai hasil uji coba klasifikasi citra *Chest X-ray* menggunakan algoritma KNN, SVM, dan CNN. Kemudian akan dijelaskan pula mengenai hasil perbandingan dari tiap algoritma tersebut.

A. Hasil Uji Coba Algoritma KNN

Tabel 6 menunjukkan hasil uji coba algoritma KNN dengan parameter *distance* dan jumlah *neighbors* yang telah ditentukan. *Dist* adalah jenis *distance*, NN adalah *Number of Neighbors* atau jumlah *neighbor*, AVG adalah rata-rata nilai *metric*, dan AVG *Perf* adalah jumlah dari rata-rata *metric accuracy*, *precision*, *recall*, dan *f1 score* dibagi dengan 4. AVG *Perf* digunakan untuk melihat rata-rata performa *metric* secara keseluruhan.

Dapat terlihat bahwa tidak ada perbedaan performa yang signifikan dengan parameter yang digunakan pada KNN. Namun dalam penelitian ini KNN dengan tipe *distance Minkowski* dan NN sejumlah 7 yang memiliki AVG *Perf* sebesar 0,921 dipilih untuk dibandingkan dengan algoritma SVM dan CNN dikarenakan memiliki AVG *Perf* terbaik.

Tabel 6. Hasil Uji Coba Algoritma KNN

Dist	NN	*Metric*	*Cross Validation*					AVG	AVG PERF
			1	2	3	4	5		
EUCLIDEAN	5	ACC	0,9177	0,9189	0,9099	0,9305	0,9228	0,92	0,9204
		PREC	0,9198	0,9198	0,9124	0,9307	0,9239	0,9213	
		REC	0,9177	0,9189	0,9099	0,9305	0,9228	0,92	
		F1SCORE	0,9182	0,9192	0,9102	0,9306	0,923	0,9202	
	7	ACC	0,9152	0,9292	0,9086	0,9266	0,9202	0,92	0,9207
		PREC	0,9197	0,9311	0,9128	0,9269	0,9215	0,9224	
		REC	0,9152	0,9292	0,9086	0,9266	0,9202	0,92	
		F1SCORE	0,9159	0,9296	0,9092	0,9266	0,9205	0,9204	
	9	ACC	0,9152	0,9292	0,9086	0,9266	0,9202	0,92	0,9204
		PREC	0,9198	0,9198	0,9124	0,9307	0,9239	0,9213	
		REC	0,9152	0,9292	0,9086	0,9266	0,9202	0,92	
		F1SCORE	0,9159	0,9296	0,9092	0,9266	0,9205	0,9204	
MINKOWSKI	5	ACC	0,9177	0,9189	0,9099	0,9305	0,9228	0,92	0,9192
		PREC	0,9198	0,9198	0,9124	0,9307	0,9239	0,9213	
		REC	0,9177	0,9189	0,9099	0,9189	0,9228	0,9177	
		F1SCORE	0,9182	0,9192	0,9102	0,9192	0,923	0,918	
	7	ACC	0,9152	0,9292	0,9086	0,9266	0,9202	0,92	0,921
		PREC	0,9197	0,9311	0,9128	0,9269	0,9215	0,9224	
		REC	0,9152	0,9292	0,9086	0,9292	0,9202	0,9205	
		F1SCORE	0,9159	0,9296	0,9092	0,9296	0,9205	0,921	
	9	ACC	0,9152	0,9292	0,9086	0,9266	0,9202	0,92	0,9207
		PREC	0,9198	0,9198	0,9124	0,9307	0,9239	0,9213	
		REC	0,9152	0,9292	0,9086	0,9292	0,9202	0,9205	
		F1SCORE	0,9159	0,9296	0,9092	0,9296	0,9205	0,921	

B. Hasil Uji Coba Algoritma SVM

Tabel 7 menunjukkan hasil uji coba pada algoritma SVM. Terdapat perbedaan performa yang cukup signifikan diantara *kernel*. Dapat dilihat bahwa algoritma SVM dengan tipe *kernel Linear* memiliki AVG *Perf* terbaik yaitu sebesar 0,930. Selanjutnya performa SVM dengan kernel *Linear* akan dibandingkan dengan algoritma KNN dan CNN.

Tabel 7. Hasil Uji Coba Algoritma SVM

Kernel	*Metric*	*Cross Validation*					AVG	AVG PERF
		1	2	3	4	5		
LINEAR	ACC	0,929	0,923	0,933	0,937	0,927	0,93	0.930
	PREC	0,93	0,923	0,934	0,937	0,937	0,932	
	REC	0,929	0,923	0,933	0,937	0,927	0,93	
	F1SCORE	0,929	0,923	0,933	0,937	0,927	0,93	
POLY	ACC	0,812	0,815	0,793	0,807	0,835	0,812	0.815
	PREC	0,825	0,821	0,808	0,836	0,836	0,825	
	REC	0,812	0,815	0,793	0,807	0,835	0,812	
	F1SCORE	0,813	0,815	0,79	0,802	0,834	0,811	
RBF	ACC	0,904	0,9	0,916	0,915	0,902	0,907	0.908
	PREC	0,906	0,9	0,917	0,915	0,915	0,91	
	REC	0,904	0,9	0,916	0,915	0,902	0,907	
	F1SCORE	0,904	0,9	0,916	0,914	0,902	0,907	

C. Hasil Uji Coba Algoritma CNN

Tabel 8 menunjukkan hasil uji coba pada algoritma CNN. Performa yang dihasilkan oleh CNN cukup baik yaitu memiliki AVG *Perf* 0,9591. Dari setiap *cross validation*, performa yang dihasilkan CNN juga cukup stabil, semua *metric* performa menghasilkan nilai di atas 0,95.

Tabel 8. Hasil Uji Coba Algoritma CNN

Metric	*Cross Validation*					AVG	AVG PERF
	1	2	3	4	5		
ACC	0,9537	0,9601	0,9537	0,9704	0,9575	0,9591	0,9591
PREC	0,9535	0,9601	0,9538	0,9711	0,9575	0,9592	
REC	0,9537	0,9601	0,9537	0,9704	0,9575	0,9591	
F1SCORE	0,9536	0,9601	0,9537	0,9703	0,9575	0,959	

D. Perbandingan Performa

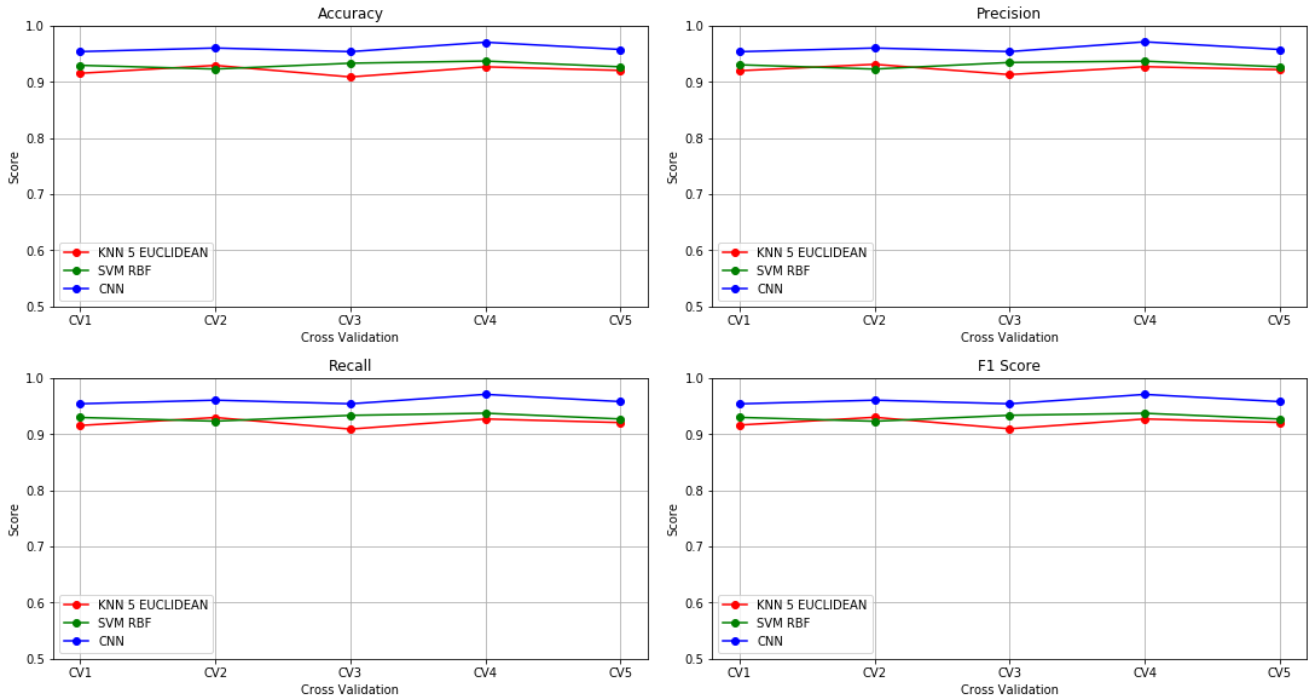
Tabel 9 menunjukkan hasil perbandingan *metric* performa dari algoritma KNN, SVM, dan CNN. Untuk algoritma KNN dan SVM menggunakan parameter yang memiliki performa terbaik seperti yang sudah dijelaskan pada subbab sebelumnya. Terlihat bahwa CNN memiliki performa yang terbaik jika dibandingkan dengan algoritma KNN dengan jumlah NN 7 dan SVM dengan kernel *Linear*. CNN memiliki *accuracy*, *precision*, *recall*, dan *f1 score* yang paling baik. Gambar 3 menunjukkan grafik perbandingan performa algoritma KNN, SVM, dan CNN.

Gap yang dimiliki antara algoritma KNN dengan SVM adalah sekitar 0,011. Sedangkan *gap* yang dimiliki antara CNN dengan KNN dan CNN dengan SVM masing-masing 0,0381 dan 0,291. Perbedaan performa antara KNN dan SVM tidak cukup jauh, namun KNN dan SVM memiliki *gap* yang cukup signifikan dengan CNN.

Tabel 9. Perbandingan Performa Algoritma

Algoritma	*Metric*	*Cross Validation*					AVG	AVG PERF
		1	2	3	4	5		
KNN 7 *MINKOWSKI*	ACC	0,9152	0,9292	0,9086	0,9266	0,9202	0,92	0,9210
	PREC	0,9197	0,9311	0,9128	0,9269	0,9215	0,9224	
	REC	0,9152	0,9292	0,9086	0,9292	0,9202	0,9205	
	F1SCORE	0,9159	0,9296	0,9092	0,9296	0,9205	0,921	
SVM *LINEAR*	ACC	0,929	0,923	0,933	0,937	0,927	0,93	0,930
	PREC	0,93	0,923	0,934	0,937	0,937	0,932	
	REC	0,929	0,923	0,933	0,937	0,927	0,93	
	F1SCORE	0,929	0,923	0,933	0,937	0,927	0,93	
CNN	ACC	0,9537	0,9601	0,9537	0,9704	0,9575	0,9591	0,9591
	PREC	0,9535	0,9601	0,9538	0,9711	0,9575	0,9592	
	REC	0,9537	0,9601	0,9537	0,9704	0,9575	0,9591	
	F1SCORE	0,9536	0,9601	0,9537	0,9703	0,9575	0,959	

Tabel 10 menunjukkan perbandingan waktu eksekusi *training* dan *testing*. Waktu eksekusi yang dihitung adalah waktu eksekusi proses training yang ditambahkan dengan proses testing pada setiap *cross validation*. Dapat dilihat bahwa CNN memiliki waktu ekskusi yang paling lama yaitu 3102,562 detik walaupun memiliki performa yang terbaik. Sedangkan SVM dengan Kernel *Linear* memiliki rata-rata waktu eksekusi yang paling cepat yaitu sebesar 32,802 detik.



Gambar 3. Grafik Perbandingan Performa KNN, SVM, dan CNN

KNN memiliki rata-rata waktu eksekusi yang tidak jauh dengan SVM *Linear* yaitu sekitar 49,49.

Tabel 10. Perbandingan Waktu Eksekusi dalam Detik

Algoritma	*Cross Validation*					AVG
	1	2	3	4	5	
KNN 7 *Minkowski*	52,56	48,50	48,61	48,81	48,97	49,49
SVM *Linear*	34,91	32,44	32,70	32,09	31,87	32,802
CNN	3154,36	3089,59	3091,79	3114,00	3063,07	3102,562

Waktu eksekusi yang lama pada CNN dikarenakan *epoch* yang digunakan cukup banyak yaitu 50. Namun berdasarkan data *history* waktu eksekusi *training* dan *testing* dari setiap *epoch*nya, CNN membutuhkan waktu rata-rata sebesar 61,5 detik dan memperoleh *accuracy* rata-rata 0,9. Hal ini sesuai dengan sifat dari *Neural Network* yang membutuhkan lebih banyak waktu namun memiliki *performance metric* yang lebih baik.

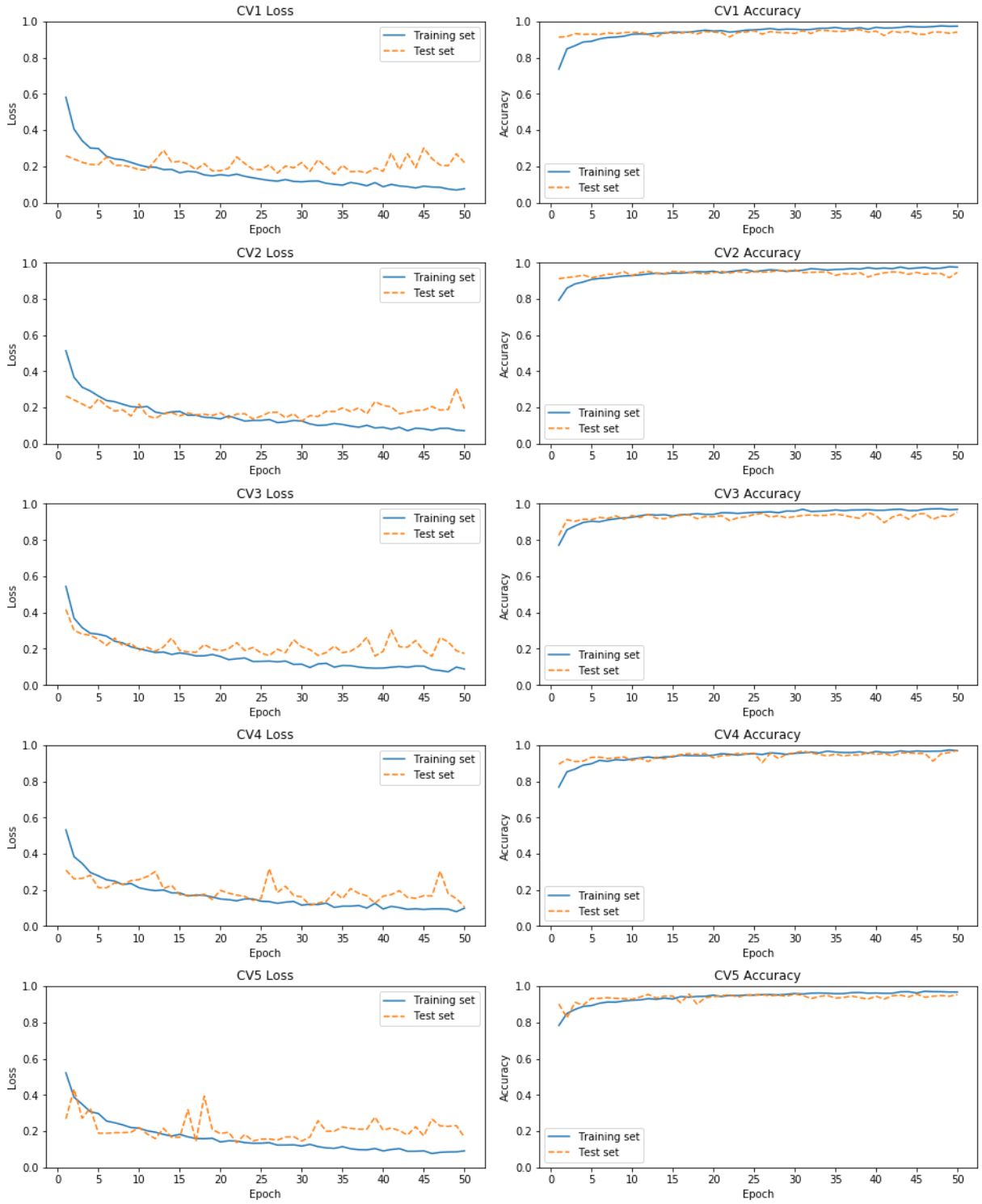
Dari model *training* dan *testing* yang digunakan pada CNN, dapat disimpulkan bahwa model yang digunakan tidak mengalami *overfitting*, karena *accuracy* yang dihasilkan pada validasi data *training* tidak jauh berbeda dengan *accuracy* yang dihasilkan saat memvalidasi data *testing*. Gambar 4 menunjukkan Grafik *Loss* dan *Accuracy* dari setiap *epoch* pada CNN.

Algoritma KNN dengan distance *Minkowski* NN 7, SVM dengan Kernel *Linear*, dan CNN tergolong cukup stabil performanya di tiap *cross validation*. Hal ini dapat dikatakan bahwa algoritma SVM, KNN, dan CNN cukup layak untuk digunakan dalam melakukan klasifikasi citra *Chest X-ray* dengan hanya menggunakan fitur intensitas *pixel*. Namun memang algoritma CNN membutuhkan waktu eksekusi pada proses *training* yang cukup lama yaitu hampir satu jam dengan jumlah data 3886 citra *Chest X-ray* yang terbagi menjadi tiga jenis citra, yaitu Normal, Covid-19, dan Viral *Pneumonia*.

Pemilihan algoritma untuk klasifikasi citra *Chest X-ray* sangat bergantung pada kebutuhan peneliti. CNN memiliki waktu yang cukup lama dalam melakukan proses *training*, namun memiliki performa yang sangat bagus. Sedangkan KNN dan SVM memiliki waktu yang relatif cepat namun performa yang dihasilkan tidak sebagus CNN. Jika peneliti memiliki spesifikasi perangkat keras atau *server cloud* yang bagus dan menginginkan performa yang bagus, waktu eksekusi CNN mungkin tidak menjadi masalah. KNN dan SVM dapat dipilih jika peneliti ingin mendapatkan model klasifikasi yang membutuhkan waktu eksekusi *training* yang relatif lebih cepat.

CNN merupakan algoritma yang cukup popular untuk klasifikasi Citra dan terbukti bahwa CNN memiliki performa yang terbaik dibandingkan KNN dan SVM untuk klasifikasi citra *Chest X-ray*. Waktu *training* yang lama hanya dilakukan di awal untuk membentuk model, sehingga tidak menjadi masalah bagi CNN jika dipilih sebagai algoritma klasifikasi citra *Chest X-ray*.

CNN dapat digunakan untuk membentuk model klasifikasi pada *dataset* citra *X-ray* dengan menggunakan *dataset training* yang berjumlah banyak. Hal ini menyebabkan waktu eksekusi *training* untuk membentuk model klasifikasi menjadi cukup lama. Saat menunggu proses pembentukan model CNN, peneliti dapat menggunakan algoritma KNN dan SVM untuk melakukan klasifikasi sementara citra *Chest X-ray* dikarenakan performa kedua algoritma tersebut masih cukup bagus. Setelah model CNN terbentuk, peneliti dapat



Gambar 4. Grafik *Loss* dan *Accuracy* CNN

menggunakan model tersbeut untuk melakukan citra *Chest X-ray.*

IV. KESIMPULAN

Berdasarkan hasil uji coba yang dilakukan dapat disimpulkan bahwa algoritma KNN, SVM, dan CNN cukup baik dalam melakukan klasifikasi citra *Chest X-ray* dengan menggunakan 5 *cross validation* pada data citra sebanyak 3886. Performa KNN dan SVM cukup bagus yaitu masing-masing memiliki akurasi 0,921 dan 0,93. Waktu eksekusi *training* KNN dan SVM juga cukup cepat yaitu masing-masing 49,49 dan 32,801 detik. CNN memiliki perofrma

akurasi lebih bagus yaitu 0,9591 namun memiliki waktu eksekusi lebih lama yaitu 3102,562.

Untuk data citra yang banyak, KNN dan SVM dapat digunakan sebagai algoritma sementara dalam melakukan klasifikasi citra *Chest X-Ray* dikarenakan waktu eksekusinya yang lebih cepat. Di saat yang bersamaan CNN dapat dilatih untuk membentuk model klasifikasi. Saat model klasifikasi CNN sudah terbentuk, maka disarankan menggunakan CNN untuk klasifikasi karena memiliki performa yang lebih bagus dibandingkan KNN dan SVM.

Rekomendasi untuk penelitian selanjutnya adalah membandingkan performa berbagai arsitektur CNN yang lain seperti *Alex Net* [11], *VGGNet* [12], *GoogLeNet* [13], *ZFNet* [14], dan *ResNet* [15] untuk mendeteksi Covid-19 menggunakan Citra *Chest X-Ray*. Selain itu penggunaan tahapan *preprocessing* dapat berguna untuk memperbaiki kualitas citra *Chest X-ray* sehingga meningkat performa metode klasifikasi.

REFERENSI

[1] Worldometers. (2021). *Coronavirus Update (Live): 86,248,818 Cases and 1,863,861 Deaths from COVID-19 Virus Pandemic - Worldometer*. Diakses dari: https://www.worldometers.info/coronavirus/ pada tanggal 5 Januari 2021.

[2] Shi, H., et.al. (2020). Radiological Findings From 81 Patients With COVID-19 Pneumonia in Wuhan, China: a Descriptive Study. *Lancet Infectious Diseases*, Vol. 20(4), pp. 425–434. DOI: 10.1016/S1473-3099(20)30086-4.

[3] Islam, N., et.al. (2020). Thoracic Imaging Tests for the Diagnosis of COVID-19. *Cochrane Database of Systematic Reviews*. DOI: 10.1002/14651858.CD013639.pub4.

[4] Yasin, R. & Gouda, W. (2020). Chest X-ray Findings Monitoring COVID-19 Disease Course and Severity. *Egyptian Journal of Radiology and Nuclear Medicine*, vol. 51. DOI: 10.1186/s43055-020-00296-x.

[5] Gao, T. (2020). Chest X-ray Image Analysis and Classification for COVID-19 Pneumonia Detection Using Deep CNN. *medRxiv*, DOI: 10.1101/2020.08.20.20178913.

[6] Abbas, A., Abdelsamea, M.M. & Gaber, M.M. (2020). Classification of COVID-19 in Chest X-ray Images Using DeTraC Deep Convolutional Neural Network. *Artificial Intelligence Applications for COVID-19, Detection, Control, Prediction, and Diagnosis*. DOI: 10.1007/s10489-020-01829-7.

[7] Deng, X., Shao, H., Shi, L., Wang, X. & Xie, T. (2020). A Classification–detection Approach of COVID-19 Based on Chest X-ray and CT by Using Keras Pre-trained Deep Learning Models. *Computer Modeling in Engineering & Sciences*, Vol. 125(2), pp. 579–596. DOI: 10.32604/cmes.2020.011920.

[8] Rahman, T. (2020). *COVID-19 Radiography Database*. Dakses dari: https://www.kaggle.com/tawsifurrahman/covid19-radiography-database pada tanggal 7 Januari 2021.

[9] Chollet, F. (2020). Keras: the Python Deep Learning API. *Keras: the Python Deep Learning API*. Diakses dari https://keras.io/ pada tanggal 18 Desember 2020.

[10] Goldman, R.N. (1991). More Matrices and Transformations: Shear and Pseudo-perspective. *Graphics Gems II*, pp. 338–341.

[11] Gonzalez, T.F. (2007). *Handbook of Approximation Algorithms and Metaheuristics*. DOI: 10.1201/9781420010749.

[12] Simonyan, K. & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-scale Image Recognition. *International Conference on Learning Representations (ICLR 2015)*.

[13] Szegedy, C., et.al. (2015). Going Deeper with Convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9. DOI: 10.1109/CVPR.2015.7298594.

[14] Zeiler, M.D. & Fergus, R. (2014). Visualizing and Understanding Convolutional Networks. *Lecture Notes in Computer Science*, vol. 8689, pp. 818–833. DOI: 10.1007/978-3-319-10590-1_53.

[15] He, K., Zhang, X., Ren, S. & Sun, J. (2016). Deep Residual Learning for Image Recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778. DOI: 10.1109/CVPR.2016.90.